Copyright
by
Ryan Mitchell Dreifuerst
2021

The Thesis Committee for Ryan Mitchell Dreifuerst
Certifies that this is the approved version of the following Thesis:

Low Resolution Sinusoid Decomposition and Estimation

APPROVED BY

SUPERVISING COMMITTEE:

Robert W. Heath Jr., Supervisor

Jeff Andrews

Low Resolution Sinusoid Decomposition and Estimation

by

Ryan Mitchell Dreifuerst

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN ENGINEERING

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2021

This work is dedicated to my family who have provided so much support through every step of the journey. To my Fiancée, Maddie, for her sacrifice for me to pursue my dreams and her resilience when I would disappear for days on end to work on the next deadline. To my parents, for encouraging me to become the person I am today, and giving me the freedom to discover who that is. To my brothers, Eric and Jacob, because “brothers are best friends”.

Acknowledgments

I wish to thank the incredible people who helped me get to this point. Thank you to my teachers and professors, who ignited a spark in me that has continued to grow. A huge thank you to my friends in Austin, Milwaukee, and across the world, who have walked with me through new languages, experiences, and challenges. Whether it is spending days in the lab or commiserating over a drink, you each contributed to where I am today. Finally, I would like to thank God for consistently reminding me that I can trust in the path He has prepared for me.

Low Resolution Sinusoid Decomposition and Estimation

Ryan Mitchell Dreifuerst, M.S.E.
The University of Texas at Austin, 2021

Supervisor: Robert W. Heath Jr.

The detection and estimation of sinusoids is a fundamental signal processing task for many applications related to sensing and communications. While algorithms have been proposed for this setting, quantization is a critical, but often ignored modeling effect. In wireless communications, estimation with low resolution data converters is relevant for reduced power consumption in wideband receivers. Similarly, low resolution sampling in imaging and spectrum sensing allows for efficient data collection. In this work, we propose SignalNet, a neural network architecture that detects the number of sinusoids and estimates their parameters from quantized in-phase and quadrature samples. We incorporate signal reconstruction internally as domain knowledge within the network to enhance learning and surpass traditional algorithms in mean squared error and Chamfer error. We introduce a worst-case learning threshold for comparing the results of our network relative to the underlying data distributions. This threshold provides insight into why neural networks tend to outperform traditional methods and into the learned relationships between the input and output distributions. In simulation, we find that our algorithm is always able to surpass the threshold for three-bit data, but often cannot exceed the threshold for one-bit data. We use the learning threshold to explain, in the one-bit case, how our estimators learn to minimize the distributional loss, rather than learn features from the data.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Detecting the number of sinusoids and estimating the amplitude, frequency, and phase is a common problem in signal processing [3–6]. In these settings, information is contained in the sinusoidal parameters of the data, i.e. in the Doppler shift from a radar signal or the angles of arrival and departure in the spatial domain. As bandwidth and array size continue to increase, e.g. in mmWave communications or wideband automotive radar, traditional systems require analog-to-digital converters (ADCs) with high resolution and rates approaching 100 Gbps or more. High resolution data converters become a limiting factor for low power consumption [7]. One solution to reduce power consumption is to reduce the resolution in ADCs, transferring the difficulty of the problem from the hardware to algorithmic design.

In principle, low resolution techniques improve efficiency–computationally, economically, or in power consumption–at the cost of introducing quantization error. For high resolution sampling the quantization error is generally modeled as an additive noise term [8, 9]. This is motivated by techniques to linearize the quantization, such as Bussgang Decomposition [10, 11]. Ultimately, the linearization has more error for 1-3 bit quantizers, which are the most desirable from a power-consumption perspective. To overcome this limitation, we investigate neural network techniques for low resolution signal processing because of their ability to learn analytical models directly from data. The use cases for such algorithms are broad, with varying compute constraints (server, cloud,

deployed), but our algorithm is primarily intended for edge computer signal processing, so we also consider important characteristics such as memory, training sample size, and execution time in our investigation. From our past work, we found convolutional neural networks were sufficiently powerful and fast for near real-time estimation of a single sinusoid [2]. As a result, our networks rely heavily on convolutional layers for parameter sharing, dimensionality reduction, and information parsing.

There is, to the best of our knowledge, little prior work that has focused directly on the joint task of detection and estimation of sinusoids from quantized data. In contrast, the field of *unquantized* detection and estimation of sinusoids has a rich history [12–17]. These techniques tend to divide the problem into separate, but related, tasks for detection and estimation. Detection is usually performed using model order estimators such as Minimum Description Length (MDL) [14,18] or Akaike Information Criterion (AIC) [13, 19]. Then, given a prediction of the number of sinusoids, the estimation is typically solved using non-parametric or parametric methods [17, 20, 21]. Alternatively, the problem can be approached using compressive sensing [22] techniques. In this paradigm, the recovered signal vector is the frequency representation of the multiple sinusoids, enabling sparse reconstruction techniques. Notable algorithms in this domain are based on the message passing algoritm [23, 24]. These methods are restricted to on-grid measurements, however, which severely affects the accuracy for limited samples. More recently, an additional class of algorithms based on neural networks have also been tasked with the same problem [25]. This has led to new state of the art results for the detection and estimation of sinusoid frequencies [25]. Neural networks have the advantage of learning directly from training data, so no assumptions regarding the SNR or model order are necessary. Instead, the model learns assumptions and distributions from the training data, possibly leading to inadvertent bias. While inductive bias is necessary for

successful learning [26], it can have adverse effects when data distribution changes. Investigation of the understandability of neural networks is an active area of research [27].

More prior work has focused on the estimation of a single quantized sinusoid in noise [28–30]. The Cramér Rao Bound was derived in [28]. Algorithms using correlation [28], dithering [29], and time-varying thresholding [30] have been proposed for quantized sinusoid estimation. Dithering and time-varying thresholding are related by effectively adjusting the sampling comparator by either a known (time-varying thresholds) or unknown factor (dithering). These methods rely on knowledge of the amplitude of the signal, or the ability to estimate by iterative grid search. Unlike the unquantized case, non-parametric methods, such as the Periodogram, are often the best estimators with extremely low-resolution sinusoid estimation [2, 29]. All of these techniques (non-parametric estimators [21], correlation-based estimators [28], dither [29] and time-varying thresholds [30]) do not make use of the quantization noise correlation [11]. This correlation is not easily modeled in accurate and tractable forms. Instead, our neural network attempts to learn the underlying relationships directly from data, bypassing the need for explicit models. We relax the assumption that the amplitude is known, thoug,h in many cases it will not be estimable either. We also contend with new concepts such as the minimum recognizable separation between signals, that strongly limits the performance of the previously mentioned estimators.

In this paper, we propose a neural architecture, SignalNet, for sinusoid decomposition and estimation from low resolution samples. Our algorithm follows traditional approaches by dividing the task into two separate networks, which we identify as the detection module and the estimation module. Detection is performed using a standard, three-layer convolutional neural network. The focal point of our algorithm is the estimation module, which sequentially estimates sinusoid parameters. Within the estimation module, we configure the network to reconstruct the input sequences

and feed the error back into the estimation module, thereby explicitly defining the relationship between the output estimates and input sequences. This method instills domain knowledge within the network and helps the model directly capture the input-output relationship for the sinusoid detection and estimation problem. We then benchmark our algorithm against classical methods for each module, as well as define and derive learning thresholds for the problem based on probability theory. In simulation, we show that both of our modules outperform traditional algorithms for quantized data. In spite of this, our detection module and amplitude estimation algorithms are not able to surpass the learning threshold for one-bit resolution data. Our contributions can be summarized as follows:

- We propose a novel deep learning algorithm for sinusoid parameter estimation. Our algorithm uses reconstruction as an internal mechanism for learning the relationship between sinusoidal parameters and input data. We show that this method is able to estimate sinusoids significantly more accurately than traditional algorithms for quantized data, while also improving the understandability of our network.

- We extend our sinusoid parameter estimator to include multiple sinusoid estimation and detection. We find that the overall network, SignalNet, is able to accurately detect the number of sinusoidal signals and their associated parameters from limited observations of quantized data. Our results shows universally improved estimation and detection performance compared to the benchmark algorithms.

- We define a new benchmark for comparing neural network algorithms and traditional algorithms based the loss function and randomness of the output distribution. The resulting threshold defines the worst-case error expected for non-adversarial data, and provides insight

into why neural networks tend to outperform other algorithms within signal processing–even for nearly random data. Applying this threshold to our simulations shows that our one-bit sinusoid detection algorithm and one-bit amplitude estimates are not able to learn meaningful input-output relationships. All other neural networks surpass the learning threshold and traditional algorithms, suggesting our algorithms are learning substantial information from the data.

Notation: $\mathbf{A}$ is a matrix, $\mathbf{a}$ and $\{a[i]\}_{i=1}^{N}$ are column vectors and a, A denote scalars. $\mathbf{A}^T$, $\overline{\mathbf{A}}$ and $\mathbf{A}^*$ represent the transpose, conjugate, and conjugate transpose of $\mathbf{A}$. The real and imaginary parts of $\mathbf{A}$ are denoted by $\mathbb{R}\mathrm{e}(\mathbf{A})$ and $\mathbb{I}\mathrm{m}(\mathbf{A})$. $A[k, \ell]$ denotes the entry of $\mathbf{A}$ in the k^{th} row and the ℓ^{th} column. Unspecified norm equations are $\|\mathbf{a}\|_2 = \mathbf{a}^*\mathbf{a}$ for vectors, and the Frobenius norm $\|\mathbf{A}\|_F = \sqrt{\mathrm{Tr}(\mathbf{A}\mathbf{A}^*)}$ for matrices. We define $j = \sqrt{-1}$. $\lfloor a \rfloor$ is the nearest integer from truncating a and $\lceil a \rceil$ is the nearest integer greater than or equal a.

Chapter 2

System model

We begin by defining the classical representation of a received sample set, comprised of multiple sinusoids, in noise. We then clarify how quantization and normalization are performed, based on common hardware considerations for gain control. Afterwards, we summarize the learning objects of each subtask and present two loss functions based on prior work and the role of our algorithm in processing signals.

An m multi-sinusoidal signal is represented by vectors of amplitudes, phases, and frequencies $\mathbf{a}, \boldsymbol{\phi}, \mathbf{f} \in \mathbb{R}^M$. The sinusoidal components are summed and sampled at sampling intervals T as

$$y[n] = \sum_{i=1}^{m} a_i \exp\left(j2\pi f_i nT + \phi_i\right). \tag{2.1}$$

Given an infinite number of samples, the exact signal components can be resolved from this model, so long as the sampling period is small enough such that the sampling theorem [31] is maintained. In realistic settings, the number of samples is finite and the desired signal is obstructed by noise $v[k]$, which is often modeled as additive, independent, and identically distributed (IID), Gaussian noise. For brevity of notation, we define the $m \times N$-dimensional matrix $\mathbf{G}$ such that $\mathbf{g}_i = \{2\pi f_i nT + \phi_i\}_{n=0}^{N-1}$. Now (2.1) can be represented as a finite set of N real and imaginary components with complex-valued noise

$$\mathbf{y} = \sum_{i=1}^{m} a_i \Big(\cos\left(\mathbf{g}_i\right) + j \sin\left(\mathbf{g}_i\right) \Big) + \mathbf{v}\,. \tag{2.2}$$

This model aligns with classic [12–17] and recent [25] approaches. We now extend the system model to include quantization effects.

Prior work with quantization has generally considered the estimation of only a single sinusoid [28,32,33] or one-bit resolution [29,30,34], thus defining the quantization levels to maximize the dynamic range (and limit clipping) is straightforward. Modeling quantization effects for multiple bit quantizers with varying signals is not as simple, and is not done in a consistent manner in literature [9, 28, 35, 36]. Yet, defining the quantization levels in a realistic manner is crucial for the system model. As a result, we define the quantization model for unit-norm power, based on traditional gain control hardware found in wireless systems. Specifically, the signal model is normalized and quantized according to a b bit uniform quantizer $\mathbb{Q}_b$ as

$$\mathbf{z} = \mathbb{Q}_b\left(N\frac{\mathbf{y}}{\|\mathbf{y}\|^2}\right) \tag{2.3}$$

$$\mathbb{Q}_b(A + jB) = \mathbb{Q}_b(A) + j\mathbb{Q}_b(B) \tag{2.4}$$

$$\mathbb{Q}_b(A) = q_i \quad \text{if } A \in (q_{i-1}, q_i]. \tag{2.5}$$

There are 2^b quantization levels $\mathbf{q} := \{q_i\}_{i=0}^{2^b}$ that can be output from $\mathbb{Q}_b$ covering 2^b+1 bins in the range of $[-1, 1]$. The output of the quantization function is assigned q_i if the input is in the range $(q_{i-1}, q_i]$, and the first and last bins extend to $\pm\infty$.

Complex number support is limited in many deep learning frameworks, so we vectorize the real and imaginary components to form the $\mathbb{R}^{N\times 2}$ received signal matrix as

$$\mathbf{X} = \Big[\mathbb{R}\mathrm{e}(\mathbf{z}), \mathbb{I}\mathrm{m}(\mathbf{z})\Big]^T. \tag{2.6}$$

Given the input to the system from (2.6), we mathematically summarize the goal of our neural network from a general learning perspective. First, we divide the problem into two tasks, detection

and estimation, to be learned by two different neural network architectures. Detection is performed first. The goal is for our algorithm to learn the parameters $\mathbf{W}_1$ for a function $\beta_{\mathbf{W}_1}$ that relates a set of observations $\mathbf{X}$ to the true number of sinusoidal components *m* in the original $\mathbf{y}$, based on a loss function $\mathrm{L}_{\mathrm{det}}$. The estimation network has parameters $\mathbf{W}_2$ and is trained to predict $\{\hat{\mathbf{a}},\ \hat{\mathbf{f}},\ \hat{\boldsymbol{\phi}}\}$, assuming perfect knowledge of *m*, with a loss function $\mathrm{L}_{\mathrm{est}}$. The training can be summarized as solve the problem

$$\mathbf{W}_1 = \underset{\mathbf{W}_i \in \mathbf{W}_{\mathrm{det}}}{\arg\min}\, \mathbb{E}_{\mathbf{X}}[\mathrm{L}_{\mathrm{det}}(m, \beta_{\mathbf{W}_i}(\mathbf{X}))] \tag{2.7}$$

$$\mathbf{W}_2 = \underset{\mathbf{W}_i \in \mathbf{W}_{\mathrm{est}}}{\arg\min}\, \mathbb{E}_{\mathbf{X}}[\mathrm{L}_{\mathrm{est}}(\{\mathbf{a},\ \mathbf{f},\ \boldsymbol{\phi}\}, \Theta_{m,\mathbf{W}_i}(\mathbf{X}))|\, m]. \tag{2.8}$$

The networks are combined and the overall estimate is produced according to

$$\hat{m} = \beta_{\mathbf{W}_1}(\mathbf{X}) \tag{2.9}$$

$$\{\hat{\mathbf{a}},\ \hat{\mathbf{f}},\ \hat{\boldsymbol{\phi}}\} = \Theta_{\hat{m},\mathbf{W}_2}(\mathbf{X}). \tag{2.10}$$

All of the outputs in (2.10) are real vectors of size $\hat{m}$. Additionally, (2.10) and (2.8) imply that the estimator $\Theta_{\hat{m}}$ is specific to the value of $\hat{m}$ provided during detection. We make use of this in Chapter 4, where internal reconstruction is used to estimate the $\hat{m}$ signal set.

We note that (2.7)-(2.10) can be applied to the traditional algorithms without quantization by reframing the training steps and meaning of $\mathbf{W}_1, \mathbf{W}_2$. By regarding the "learning" phase as a period of collecting sample statistics, (2.7) and (2.8) can be thought of as a tuning process for determining Bussgang gain [11] and algorithm parameters such as dither scaling [29]. The minimization is performed using known equations, such as the multi-level quantization presented in [36]. Then the functions β, θ can be regarded as the combination of Bussgang Decomposition, a detection algorithm (e.g. AIC, MDL) and an estimation algorithm (non-parametric, dithering, etc).

The goal of our network is to estimate the relevant sinusoidal components as an initial signal processing task, leaving possible filtering or decoding to subsequent algorithms. As a result, the detection algorithm should learn to overestimate the number of signals during uncertainty, rather than underestimate. This way higher level algorithms can filter out small or irrelevant components rather than miss signals altogether. To instill this knowledge, we define a heavy-sided loss function for the detection network shown in Fig. 2.1 and defined as

$$\mathrm{L}_{\mathrm{det}}(m, \hat{m}) = \begin{cases} e^{m-\hat{m}} - 1 & \text{if } m \geq \hat{m} \\ \frac{1}{2}(m-\hat{m})^2 & \text{otherwise.} \end{cases} \tag{2.11}$$

The specific internal functions are chosen to be differentiable, smooth, and ensuring that the loss has the relationship $\mathrm{L}_{\mathrm{det}}(m, m+1) < \mathrm{L}_{\mathrm{det}}(m, m-1) < \mathrm{L}_{\mathrm{det}}(m, m+2)$, which we make use of in our analysis of the learning threshold. Assuming learnability and sufficient smoothness, gradient-based optimization should learn to estimate $\hat{m} \in (m-1, m+2)$ in expectation.

In the estimation module, we will train the network based on mean squared error (MSE), assuming the true number of signals m is known. Then, when both modules are combined, we evaluate the network according to the Chamfer distance [37]

$$L_{\text{chamfer}} = \sum_{f_i \in \mathbf{f}} \min_{\hat{\mathrm{f}}_k \in \hat{\mathbf{f}}} |f_i - \hat{\mathrm{f}}_k| + \sum_{\hat{\mathrm{f}}_i \in \hat{\mathbf{f}}} \min_{f_k \in \mathbf{f}} |\hat{\mathrm{f}}_i - f_k|. \tag{2.12}$$

The comparison can be applied to the frequency estimates as shown, or any of the desired quantities. Effectively, this loss function increases for detecting the incorrect number of signals $\hat{m} \neq m$ and their associated values, although poor estimates are penalized twice, due to the two summations. The network is not retrained on this loss for two reasons: 1) it is closely related to the MSE if the number of signals detected is correct, and 2) because gradient propagation is not possible with the discrete selection. We discuss these considerations further in Chapter 5.

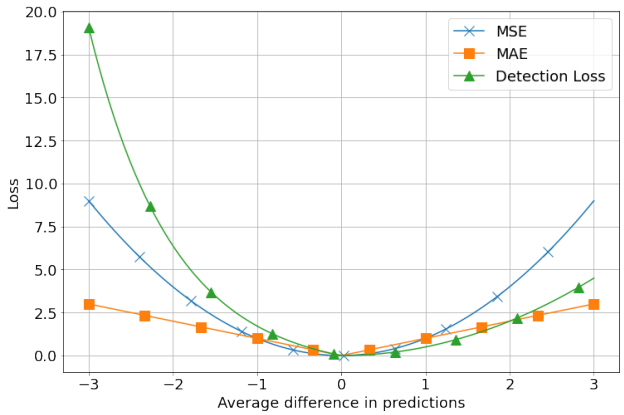


Figure 2.1: Graphical comparison of our proposed detection loss with mean squared error (MSE) and mean absolute error (MAE). Detection loss is a heavy-sided loss function suggesting that the network should generally over estimate the spectral components in a signal.

Chapter 3

Learning Threshold

Before proposing our algorithm, we first introduce a baseline for algorithmic comparison, particularly for neural network evaluation. Because machine learning algorithms learn from data, there are inherent constraints and underlying distributions that algorithms can optimize for, creating unfair bias when compared to traditional algorithms. For this reason, we also define a learning threshold based on statistical estimation theory and Empirical Risk Minimization [26]. First, let $\mathbf{x} \in \mathbb{R}^p$ be the input and $y \in \mathbb{R}$ be the true output, with joint distribution $P(\mathbf{x}, y)$. The goal is for an estimator, $g_\theta(\mathbf{x})$ that is defined by parameters θ, to predict y. We wish to minimize a loss function, and for simplicity we start with mean squared error (MSE), for K predictions. Note, MSE is defined as $L(y, f_\theta(\mathbf{x})) = 1/K \sum_{i=1}^{K} (y_i - g_\theta(\mathbf{x}_i))^2$, where the parameters θ are updated based on training data. This leads to the classical formulation

$$\min_\theta L(y, g_\theta(\mathbf{x})) = \min_\theta \mathbb{E}_{\mathbf{x}} \, \mathbb{E}_{y|\,\mathbf{x}}[(y - g_\theta(\mathbf{x}))^2 |\, \mathbf{x}] \tag{3.1}$$

$$g_\theta(\mathbf{x}) = \mathbb{E}[y|\,\mathbf{x}]. \tag{3.2}$$

Now, we define the learning threshold as the worst-case training error, in the non-adversarial case, that occurs when X is independent from Y. This simplifies (3.1)-(3.2) to

$$L_{\text{threshold}}(y, g_\theta(\mathbf{x})) = \min_\theta \mathbb{E}_y[(y - g_\theta(\mathbf{x}))^2] \tag{3.3}$$

$$g_\theta(\mathbf{x}) = \mathbb{E}[y] \quad \forall\, \mathbf{x}\,. \tag{3.4}$$

The result, for this simplified example, is that the estimator should simply estimate the mean of the output variable, and the resulting error will be the variance of Y. This well-known relationship is a simplification of our learning threshold, and is directly used for our algorithms trained with MSE loss. It is similar to other established estimation techniques such as the minimum variance unbiased estimator or the minimum mean squared error estimator when the loss function is the mean squared error as we have shown. The formulation extends beyond mean squared error though, and can be applied to more interesting cases, in particular the detection module with loss function defined by (2.11). This distribution is heavy-sided and the output variable can only take integer values, leading to a more rigorous analysis. The threshold analysis is not just an interesting derivation however, in fact, we show in Chapter 8 that our algorithms for both detection and amplitude estimation converge to this threshold for one-bit data.

The learning threshold will show that there is an inherent bias that is observable just from the distributions and loss function that can often surpass traditional methods. The threshold is relevant for two reasons: 1) it provides a scalar metric for the uncertainty in the output variable, with respect to the loss function, and 2) comparing the neural network's loss with this threshold provides clarity about whether a neural network is learning features or blindly optimizing for distributions. In general, we expect our neural network estimators to outperform the learning threshold, suggesting some relationship is learned between the input and the output. Because the quantization function is not independent from the noise however, low SNR data can be misleading or adversarial. In contrast, the threshold is independent of the input SNR, so we would expect to see results that worse than the learning threshold for sufficiently distorted signals. After defining our algorithms, we will derive learning thresholds for the detection and estimation problems based on our simulation setting to lend some intuition to why neural networks outperform traditional methods.

Chapter 4

SignalNet

In this chapter, we define the SignalNet architecture, comprised of two sub-networks for detection and estimation. We focus heavily on the estimation side of the problem, because extraneous signals can be eliminated at later points, but poorly estimated signals are detrimental to application-specific information. To improve the learning of our estimation module, we include domain knowledge through reconstruction and cancellation, similar to successive interference cancellation [38]. We provide a description, figures, and a table summarizing the operation of SignalNet, and source code will also be made available at [39].

We design SignalNet, shown by the modules and overall architecture in Figures 4.1-4.3, according to the general structure in (2.6)-(2.10), where one module detects the number of sinusoids in the signal and another estimates the sinusoids parameter from an observed sequence $\mathbf{X}$. The detection module is used to estimate the number of sinusoids present in a signal, and is modeled using a three layer convolutional neural network. The sinusoid estimation module, however, requires more knowledge of the relationship between the output parameters and the input $\mathbf{X}$ to effectively capture the information. To do this, we design the estimator around the idea of successive-estimation and cancellation, similar to interference cancellation methods in non-orthogonal communications [38]. The network architecture first estimates the parameters for a single sinusoid, reconstructs the time-domain signal, and compares it with the original input. Then, it estimates the

Table 4.1: Network parameters for one detection module and sinusoid estimator.

Network	**Layer**	**Parameter**	**Activation**	**Output Dimension**
Detection Module	Conv + Pooling + BN	32 Filters	ReLU	(32, 32, 32)
Detection Module	Conv + Pooling + BN	64 Filters	ReLU	(32, 16, 64)
Detection Module	Conv + Pooling + BN	128 Filters	ReLU	(32, 4, 128)
Detection Module	Dropout	0.7 Rate		(32, 512)
Detection Module	Fully Connected	128 Neurons	ReLU	(32, 128)
Detection Module	Fully Connected	64 Neurons	ReLU	(32, 64)
Detection Module	Fully Connected	5 Neurons	Softmax	(32, 5)
Sinusoid Estimator	Conv + Pooling + BN	32 Filters	ReLU	(32, 32, 32)
Sinusoid Estimator	Conv + Pooling + BN	64 Filters	ReLU	(32, 16, 64)
Sinusoid Estimator	Fully Connected	128 Neurons	SeLU	(32, 128)
Sinusoid Estimator	Fully Connected	32 Neurons	SeLU	(32, 32)
Sinusoid Estimator	Fully Connected	$\hat{m}$ Neurons		(32, $\hat{m}$)
Sinusoid Estimator	Conv + Pooling	16 Filters	ReLU	(32, 32, 16)
Sinusoid Estimator	Conv + Pooling	32 Filters	ReLU	(32, 16, 32)
Sinusoid Estimator	Fully Connected	64 Neurons	SeLU	(32, 64)
Sinusoid Estimator	Fully Connected	$\hat{m}$ Neurons		(32, $\hat{m}$)

parameters for the next sinusoid from the difference. This knowledge helps our neural architecture better handle multiple frequencies and is intuitive. A block diagram of the sinusoid estimator is shown in Figure 4.2. To perform the reconstruction, our sinusoid estimator outputs the amplitude and frequency of the sinusoid and then regenerates the N length samples that sinusoid would produce. Internally, each sinusoid estimator has two branches of two convolutional layers where one branch has batch normalization for estimating frequency and phase components and the other does not to retain amplitude information. Hyperparameters are tuned using iterative grid search for both the sinusoid estimator and detection module, and layer specifics can be found in Table 4.1.

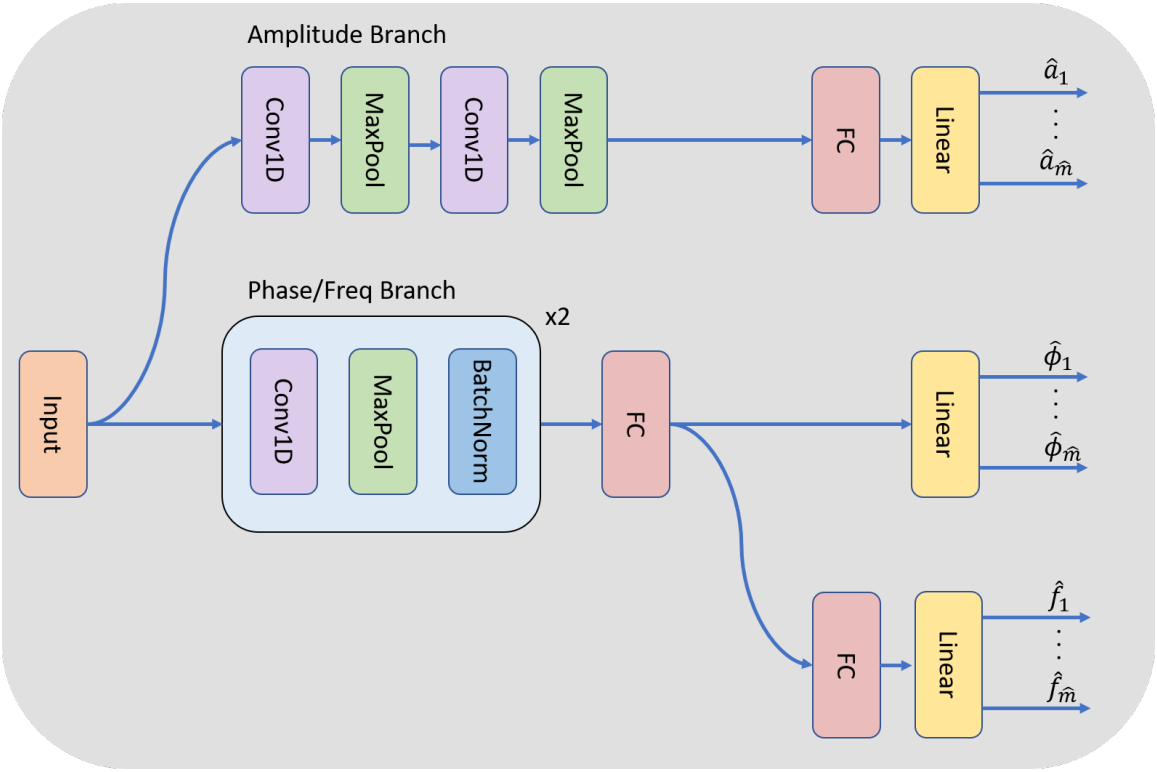


Figure 4.1: A diagram of a block estimator used to determine the parameters for a specific number of sinusoids. The two path network is made up of an unnormalized branch for amplitude estimation and a normalized branch for frequency and phase estimation. Frequency and phase estimation are linked because of the natural linear relationship between frequency and change in phase over a set of N samples.

Recalling (2.10), we train all of the networks independently, where each $m \in M$ possible number of outputs and each quantization resolution $b \in B$ result in a different network, totaling $M \times B$ networks in our investigation. In this setup, we build the a b-bit SignalNet architecture from M different sinusoid estimator networks. The need for having discrete estimators for each number of sinusoids is based on two considerations: real-time feasibility and our choice of internal reconstruction. The real-time feasibility is due to graph-based optimization, which does not allow

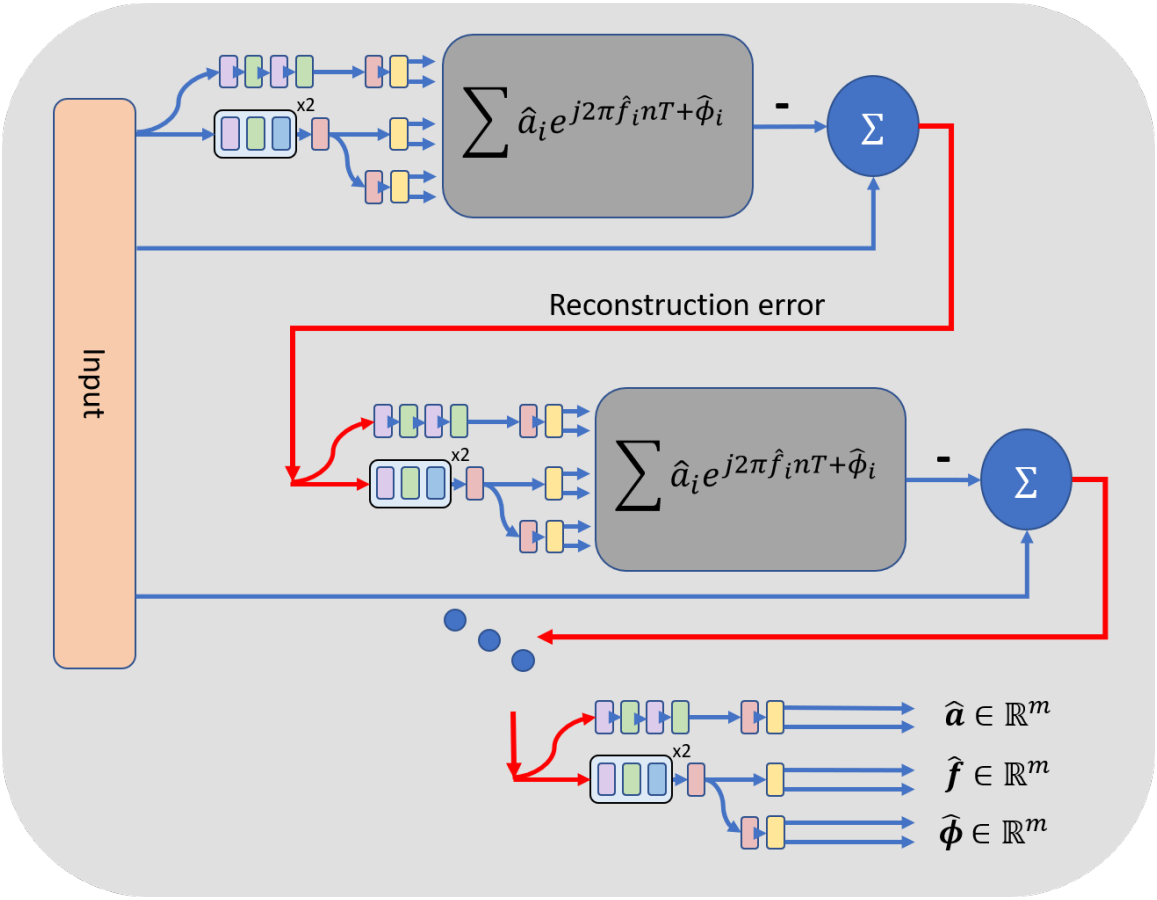


Figure 4.2: The sinusoid estimator is comprised of multiple block estimators. These networks successively estimate $\{1, \ldots m\}$ sinusoids by estimating and reconstructing each number of sinusoids and recomputing on the error. Internal reconstruction is used to provide clear relationships between the estimated outputs $[\mathbf{a}, \mathbf{f}, \boldsymbol{\phi}]$ and the input $\mathbf{X}$. This formulation results in each $m \in 1, \ldots M$ estimator producing different outputs and learning different features.

for dynamically-sized outputs without sacrificing significant speed, especially for inference. Our focus is on efficient machine learning for signal processing, so near real-time inference is critical. In addition to computational performance, our use of internal reconstruction is reliant on having separate estimators for the number of sinusoids for good performance.

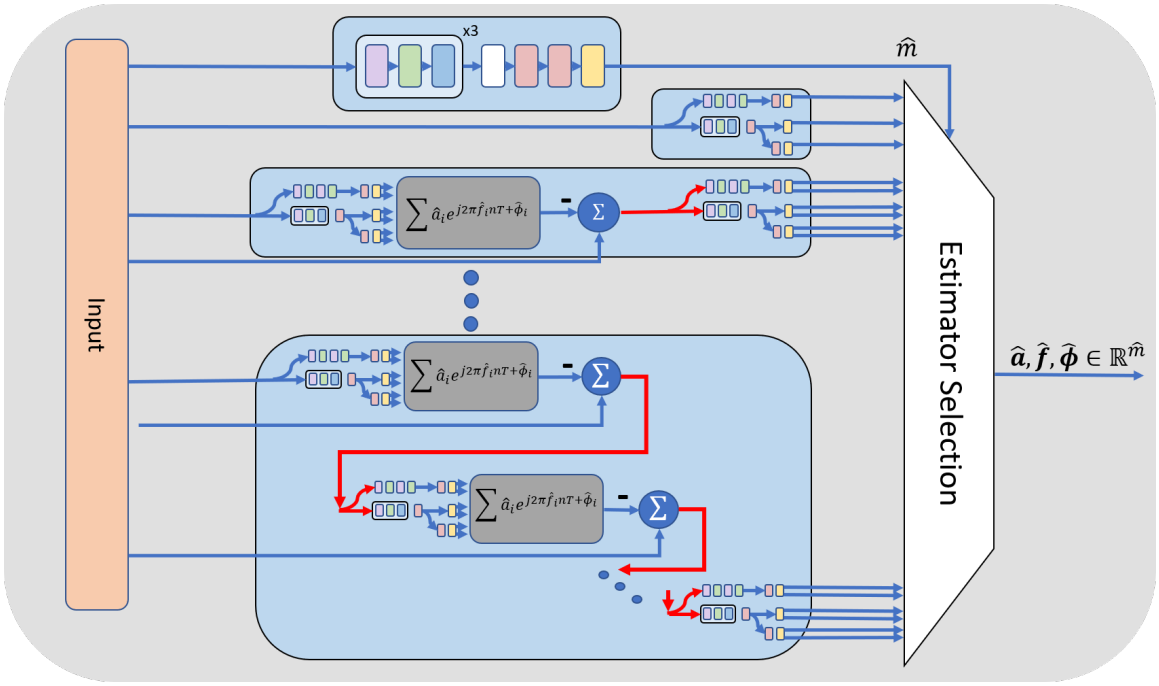


Figure 4.3: A block diagram of SignalNet. The architecture includes M sinusoid estimator modules and one detection module, all trained for a specific quantization resolution b. Each subnetwork is highlighted, with the top nextwork being the detection module, and the following networks are the $\{1, 2, \text{and} M\}$ sinusoid estimators. In our investigation, we develop a SignalNet variant for one-bit and three-bit resolutions, resulting in $(M + 1) \times 2$ total networks. Distillation [1] could also be considered to overcome the need for specific variants, but we leave this for future work.

When estimating and reconstructing an unknown number of sinusoids, the results have different frequencies and amplitudes depending on the anticipated number of components present in the signal. In other words, our network does not just find the peaks of DFT bins. Instead, it finds the signal most closely approximating the input, while gradients are only updated from the parameter error. This is best understood from the visualization in Fig. 4.4, where we show a two sinusoid signal with the best reconstruction using $\{1, 2, 3\}$ sinusoids. The minimal error for the single sinusoid estimate will not be either of the two sinusoids, but some sinusoid between the two.

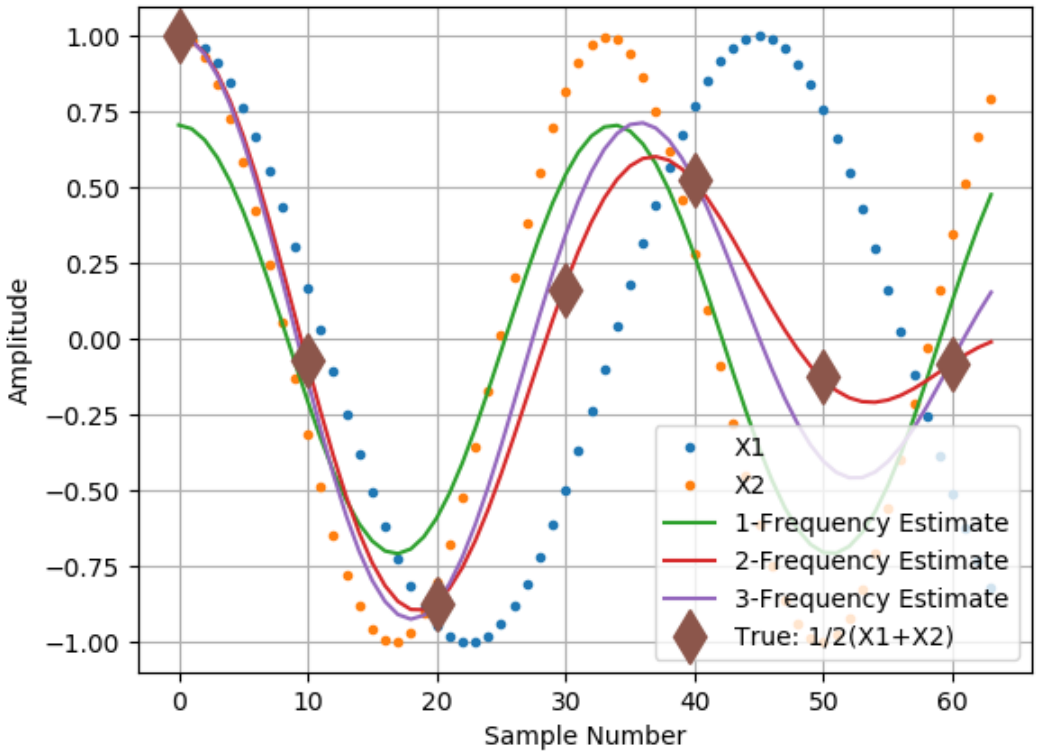


Figure 4.4: An illustration of the internal reconstruction task using values from the estimates of SignalNet. Although the network is trained based on the mean squared estimation error, the internal layers seek to most closely represent the signal for a given $\hat{m}$. In the figure, neither X1 nor X2 are output when using only one sinusoid, shown in green. Instead, the best estimate is that which reconstructs the sinusoid created by the brown diamonds. Because the true signal was two sinusoids, the 2-Frequency Estimate performs best and aligns with the true signal shown by the diamond markers.

After training, the corresponding detection and estimation modules are joined according to Figure 4.3.

Chapter 5

Simulation setup

We consider a simulation setup similar to [25], with the notable exception of considering a reduced range of sinusoids $M = 5$. The key most important aspect is the non-uniform frequency distribution of our data generation, as outlined in Algorithm 1. In this setting, a random number of sinusoids is selected. Then the uncertainty in the spacing is selected by a folded normal distribution along with the initial frequency offset from a uniform distribution. Finally, the maximum frequency is compared with 0.5, to ensure that no frequency content is undersampled according to the Nyquist criterion [31]. The frequency distribution for each *m* is shown in Fig. 5.1 and the overall distribution is shown in Fig. 5.2. Then amplitudes and phases are drawn from uniform distributions and the signal is constructed by summing over all of the sinusoids and adding Gaussian noise to obtain $\{y\}_{n=0}^{N-1}$ for a desired SNR. In the final steps, the signal is normalized to unity power norm, representative of an automatic gain controller or similar control systems at the input to the data converter. Finally, quantization $\mathbb{Q}_b$ is applied according to the number of bits b.

We select this setup based on two considerations for the difficulty of the problem: the length of the data and the separation of the frequencies. For length N discrete observations, resolving unquantized data with separation of $1/N$ is nearly impossible in the presence of noise [40,41]. Although off-grid techniques can be used to resolve sparse signals (for some appropriate basis), resolving two signals with less separation and in the presence of noise is effectively not possible [41].

We do not strictly enforce this separation, but use the folded normal distribution to discourage it. This leads to estimators that may have overlapping frequency content that could represent interference. Additionally, the amplitude of the signal directly contributes to the difficulty of the problem, so we constrain the amplitudes to the range of $[0.1, 1.0]$. These choices create a difficult simulation setup, as seen by the simulation results in Chapter 8, especially the amplitude estimation results in Figure 8.3.

Algorithm 1 Simulation Data Generation

Draw a random integer, m from $[0,\ M-1]$
do
    $w_0 \sim \mathcal{U}(0, 0.25)$
    $f_1 \leftarrow w_0$
    for $i \in [1, ...m-1]$ **do**
        $w_i \sim |\mathcal{N}(0, 2.5/N)|$
        $f_{i+1} \leftarrow w_0 + i/N + w_i$
    end for
while $\max f_i > 0.5$
$a_i \sim \mathcal{U}(0.1, 1.0) \quad i \in [1, ...m]$
$\phi_i \sim \mathcal{U}(0, 2\pi) \quad i \in [1, ...m]$
$\mathbf{y} \leftarrow \sum_{i=0}^{m} a_i \exp(2\pi j f_i n + \phi_i) + \mathbf{v}$
$y_i \leftarrow \frac{y_i}{\|\mathbf{y}\|} \quad i \in \{0, 1, ..N-1\}$
$\mathbf{x}_i \leftarrow \mathbb{Q}_b(y_i)$

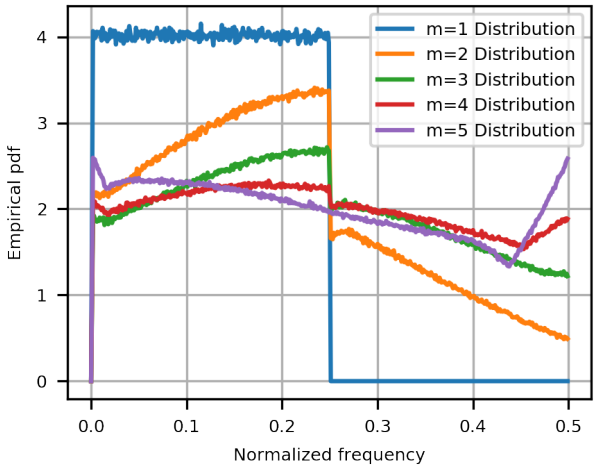


Figure 5.1: Distribution of frequency content $\mathbf{f}$ over $10,000,000$ trials. The density functions show how the changing value of m strongly affects the frequency distribution, causing significantly more challenging results as m increases. It is clear this distribution, with knowledge of m, contains information that can be leveraged by our neural network.

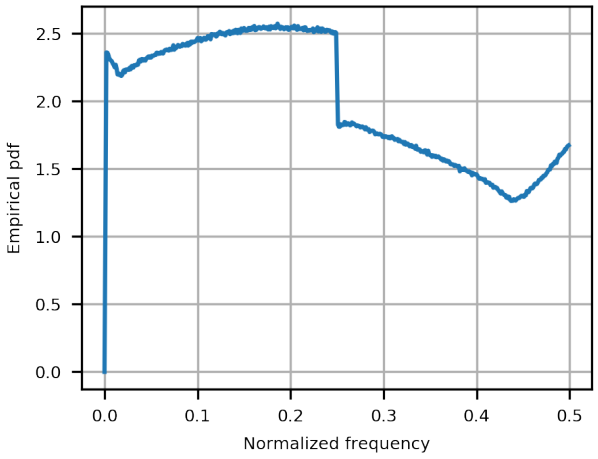


Figure 5.2: The overall frequency content distribution. Because each value of $m \in M$ are uniformly likely, the average over all of the frequencies is the point-wise average from Figure 5.1. This distribution would be difficult to analytically leverage in classical systems, which is one reason deep learning is often surpassing traditional algorithms in simulated scenarios.

Chapter 6

Benchmarks

Because there are, to the best of our knowledge, no other algorithms designed for sinusoidal decompositition and estimation from quantized data, we limit the benchmarks to well-established estimators, in combination with Bussgang Decomposition [11]. We first highlight the two common detection methods, AIC and MDL and the limitations that minimizing model order has on the detection loss function, (2.11). We also investigate EM-VAMP, a message passing algorithm, as a method for determining the model order in a compressed sensing form. Then we suggest the non-parametric methods we compare against based on the Fast Fourier Transform (FFT) and Periodogram. We conclude the chapter with a brief recount of Bussgang Decomposition and how it applies in our system model.

We evaluate our detection module against AIC and MDL, although both of these methods are suboptimal due to misaligned goals. Fundamentally, these methods attempt to minimize the complexity while capturing the number of spectral components. In contrast, the loss function (2.11), penalizes under-estimates more than over-estimates. This leads to significantly underperforming detection results for AIC and MDL, so we double the bit resolution for these methods to show more comparable results. We also show the EM-VAMP method, using a learned threshold for determining the model order from the recovered vector support. While not following the exact same structure as we outlined in Chapter 2, this benchmark achieves the best error at low SNR,

although is not competitive at high SNR. The EM-VAMP method recovers the frequency-grid-aligned sparse signal (the sinusoidal components in the frequency domain) from a 2x oversampled DFT observation matrix $\mathbf{A}$. Mathematically, the EM-VAMP method works to minimize the error in

$$\mathbf{y} = sign(\mathbf{A}\,\mathbf{x} + w) \tag{6.1}$$

by recovering the sinusoidal amplitudes and phases in the sparse coefficients of $\mathbf{x}$. This framework is limited due to its on-grid nature, however, so we only consider it for detection, rather than detection and estimation. For more information on EM-VAMP and GAMP methods, see [23, 24].

To evaluate our multi-sinusoid estimator, we employee the Periodogram, because it has been shown to produce better estimates for low resolution sinusoidal data than eigendecomposition and dithering methods [2, 29]. The Periodogram is calculated from the scaled-and-squared, zero-padded Fast Fourier Transform (FFT) with $N_0 = 2^{16}$ to ensure that grid resolution is not a limiting factor. Although amplitude and frequency information can be obtained from the Periodogram, the phase information is recovered from the inverse tangent of the ratio between the imaginary and real components of the FFT. Precisely, the phase estimate is calculated as

$$\mathbf{r} = \text{FFT}([\mathbf{x}, \mathbf{0}_{N_0-N}]) \in \mathbb{C}^{N_0} \tag{6.2}$$

$$\text{Peaks} = m\text{-argmax}\,\mathbf{r} \tag{6.3}$$

$$\hat{\boldsymbol{\phi}}_{\text{FFT}} = \arctan\left(\frac{\mathbb{I}\text{m}(\mathbf{r}[\text{Peaks}])}{\mathbb{R}\text{e}(\mathbf{r}[\text{Peaks}])}\right). \tag{6.4}$$

The peak finding algorithm in (6.3) simply finds the m largest local maxima by three-point neighbor comparison of the oversampled N_0 FFT. Prior to applying any of the benchmark algorithms, we also make use of the Bussgang Decomposition to reduce quantization effects.

The Bussgang Decomposition is a method for linearizing a function by computing the linear minimum mean squared error estimator. In the case of quantization, it has been shown to be a useful tool to separate the signal and quantization noise into two uncorrelated terms [11, 36, 42] assuming a gaussian signal. The essence of the decomposition is to calculate a gain factor G such that, for a nonlinear function $y = U(x)$, the result is

$$\begin{aligned} y &= Gx + \eta \\ \mathbb{E}[y\eta] &= 0 \\ \mathbb{E}[x\eta] &= 0 \end{aligned}$$

which is the linear minimum mean squared error estimator (LMMSE) of x from y. This formulation extends to vector signals, causing G to become $\mathbf{G}$ which is a diagonal matrix. The exact formulation for multi-bit quantization, $U = \mathbb{Q}_b$, is provided in [36]. One of the key components in the model is knowledge of the first and second moments of $\mathbf{x}$. Because of our power normalization and gaussian noise, we observe that the complex observation vector $\mathbf{X}[:, 0] + j\mathbf{X}[:, 1] \sim \mathbb{CN}(\mathbf{0}, \mathbf{1})$. Then, by using the inverse of $\mathbf{G}$, which is simple due to the diagonal structure of $\mathbf{G}$, and neglecting the distortion, we can obtain the linearized recovery of $\mathbf{x}$

$$\tilde{\mathbf{x}} = \mathbf{G}^{-1}(\mathbf{X}[:, 0] + j\mathbf{X}[:, 1]). \tag{6.5}$$

We apply the previously described benchmark detection and estimation algorithm to $\tilde{\mathbf{x}}$ in our simulation results. We found this improved the benchmark algorithms by $1 - 3$dB in most cases.

Chapter 7

Model Training

In our setup, we train the two modules, the detection module and the sinusoid estimation module, separately. We employ the loss function in (2.11) to train the detection module for 20 epochs with $50,000$ realizations of one-hot encoded signal counts. We evaluate the estimation module using the MSE in the training data, assuming that the true number of signals is known. This way each amplitude-frequency-phase estimate corresponds to a true set of parameters. We then train on $100,000$ data samples, using gradient updates from a scaled sum of loss functions based on the learning thresholds. We define this scaling to achieve a normalized loss as

$$\boldsymbol{\ell} = \left[L(\mathbf{a}, \hat{\mathbf{a}}), L(\mathbf{f}, \hat{\mathbf{f}}), L(\boldsymbol{\phi}, \hat{\boldsymbol{\phi}}) \right]^T \tag{7.1}$$

$$\boldsymbol{\ell}_{\text{threshold}} = \left[\frac{1}{\mathrm{L}_{\text{threshold},\mathbf{a}}}, \frac{1}{\mathrm{L}_{\text{threshold},\mathbf{f}}}, \frac{1}{\mathrm{L}_{\text{threshold},\boldsymbol{\phi}}} \right]^T \tag{7.2}$$

$$\ell_{\text{eff}} = \frac{1}{m} \boldsymbol{\ell}^T \boldsymbol{\ell}_{\text{threshold}}. \tag{7.3}$$

This novel scaling is to prevent the gradients from being dominated by parameters that have larger variance i.e. the true phase values are drawn from a much larger range than the amplitude or frequencies. We will also use (7.3) again for computing the overall chamfer loss as a unified metric for SignalNet. The specific learning thresholds are derived in Chapter 8. While training, we employ learning rate reduction and early stopping based on a separate validation set loss. When evaluating

our algorithms, we generate $8,000$ samples of new data following the same algorithm for our test data.

Before evaluating our algorithm, we briefly remark on data size and memorization. The size of the training dataset, while initially appearing large, is extremely small versus the input data space. For example, in the smallest case, $b = 1$, the size of the input data space is 2^{2N}, where $N = 64$ and there is a real and imaginary component, each taking a value of ± 1. Similarly, the size of our neural networks is much smaller than the data dimension as well, with the total number of parameters for SignalNet being 2.7M parameters, but only at most 0.9M parameters in any individual network. As a result, our algorithm is not capable of memorizing the data, and could likely benefit from increasing the dataset size by orders of magnitude. Our results in Chapter 8 show the efficacy of our algorithm to learn under this strict data constraint. Furthermore, near real-time inference is achievable with modern processing hardware. With a Tesla P100 GPU and limited optimization, we were able to achieve inferences in under $200\mu s$.

Chapter 8

Simulation results

In this chapter, we evaluate each SignalNet component, first individually, then as a whole. We derive the learning threshold for each subtask, shown by dashed lines in each plot, and compare the simulation results of our algorithm along with the benchmark traditional algorithm. In the final setting, we combine the two modules and compare it with different combinations of our modules and the traditional methods.

8.1 Detection

First, we evaluate the detection module, which is compared with the AIC and MDL algorithms in Fig. 8.1. In this situation, the loss function is the detection loss defined in (2.11), and the possible outputs are $\hat{m} \in \{1, 2, ..5\}$. Then, the learning threshold is defined as

$$\mathrm{L}_{\text{threshold},m}(m, g_\theta(\mathbf{X})) = \min_{\hat{m}} \mathbb{E}_m[\mathrm{L}_{\mathrm{det}}(m, \hat{m})] \tag{8.1}$$

where $\hat{m} = g_\theta(\mathbf{X}) \; \forall \; \mathbf{X}$ is a constant estimator. Because the function is smooth, differentiable, and convex, it is a simple step to go from (8.1) to solving for $\hat{m}$ from

$$0 = \frac{\partial}{\partial m} \mathbb{E}_m[\mathrm{L}_{\mathrm{det}}(m, \hat{m})]. \tag{8.2}$$

Then, because $\mathrm{L}_{\mathrm{det}}$ is bounded over the inputs, the derivative and expectation can be interchanged by the bounded convergence theorem. Following from (8.2),

$$0 = \mathbb{E}_m \left[\frac{\partial}{\partial m} \mathrm{L}_{\mathrm{det}}(m, \hat{m}) \right] \tag{8.3}$$

$$= \frac{1}{M} \sum_{m=1}^{M} \frac{\partial}{\partial m} \mathrm{L}_{\mathrm{det}}(m, \hat{m}). \tag{8.4}$$

The step from (8.3) to (8.4) is because m is uniformly distributed and discrete, so the expectation is the arithmetic mean. In our definition of $\mathrm{L}_{\mathrm{det}}$, we made the important restriction that $\mathrm{L}_{\mathrm{det}}(m, m+1) < \mathrm{L}_{\mathrm{det}}(m, m-1) < \mathrm{L}_{\mathrm{det}}(m, m+2)$, so the resulting threshold estimator will be $\hat{m} \in (\lceil M/2 \rceil, \lceil M/2+1 \rceil)$ for our simulation setting. We now ignore the constant $1/M$ factor, replace $\frac{\partial}{\partial m} \mathrm{L}_{\mathrm{det}}$ with its derivative components, and solve for the threshold estimator

$$0 = \sum_{m=1}^{\lfloor \hat{m} \rfloor} (m - \hat{m}) + \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m-\hat{m}} \tag{8.5}$$

$$= \frac{1}{\lfloor \hat{m} \rfloor} \sum_{m=1}^{\lfloor \hat{m} \rfloor} (m) - \hat{m} + \frac{1}{\lfloor \hat{m} \rfloor} \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m-\hat{m}} \tag{8.6}$$

$$= \frac{\lfloor \hat{m} \rfloor + 1}{2} - \hat{m} + \frac{1}{\lfloor \hat{m} \rfloor} \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m-\hat{m}}. \tag{8.7}$$

Defining $\alpha = \frac{\lfloor \hat{m} \rfloor + 1}{2}$ simplifies the results to achieve a expression for the constant estimator:

$$0 = \alpha - \hat{m} + \frac{e^{-\hat{m}}}{\lfloor \hat{m} \rfloor} \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m} \tag{8.8}$$

$$= -e^{\hat{m}}(\hat{m} - \alpha) + \frac{1}{\lfloor \hat{m} \rfloor} \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m} \tag{8.9}$$

$$= -e^{\hat{m}-\alpha}(\hat{m} - \alpha) + \frac{1}{\lfloor \hat{m} \rfloor} \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m-\alpha} \tag{8.10}$$

$$\hat{m} = W\left(\frac{1}{\lfloor \hat{m} \rfloor} \sum_{m=\lceil \hat{m} \rceil}^{M} e^{m-\alpha}\right) + \alpha \tag{8.11}$$

$$= W\left(\frac{1}{\lceil M/2 \rceil} \sum_{m=\lceil M/2+1 \rceil}^{M} e^{m-\alpha}\right) + \alpha. \tag{8.12}$$

Here, $W(\dots)$ is the Lambert W function, and the final expression comes from the choice of $\mathrm{L}_{\mathrm{det}}$ and $\hat{m} \in (\lceil M/2 \rceil, \lceil M/2+1 \rceil)$. Evaluating for $M = 5$ leads to $\hat{m} \approx 3.68995$, $\mathrm{L}_{\text{threshold},m} \approx 1.67$. We note that although the definition of m and $\hat{m}$ must be integer values, fractional amounts can be effectively obtained by choosing between the floor and ceiling values with the appropriate probability, irrespective of the input. In this setting, the threshold estimator is not truly constant, but the estimator does not depend on the input. The threshold is shown in Figure 8.1 by the dashed line and provides a metric for the maximum expected error if the noise and quantization effects are negligible.

Evaluating the estimators, we first start with the simplest consideration: can the estimators perform better than a blind estimator? Most importantly, all of the estimators fail to reach the threshold for SNR < -5dB, showing that the noise and quantization effects cause the algorithms to perform worse than if the data $\mathbf{X}$ was ignored. Additionally, the one-bit detection module is never able to surpass the threshold, suggesting that the multiple sinusoid detection algorithm is not able to learn meaningful results from one-bit data. The three bit results are able to improve upon the threshold for SNR > -4dB, and perform better than higher resolution versions of AIC and MDL across the entire SNR range. Further, it can be observed that extremely low SNR data with greater variation (higher bit resolution) results in worse estimators. This validates the intuition that low SNR, coarsely quantized data is effectively adversarial compared to higher SNR data.

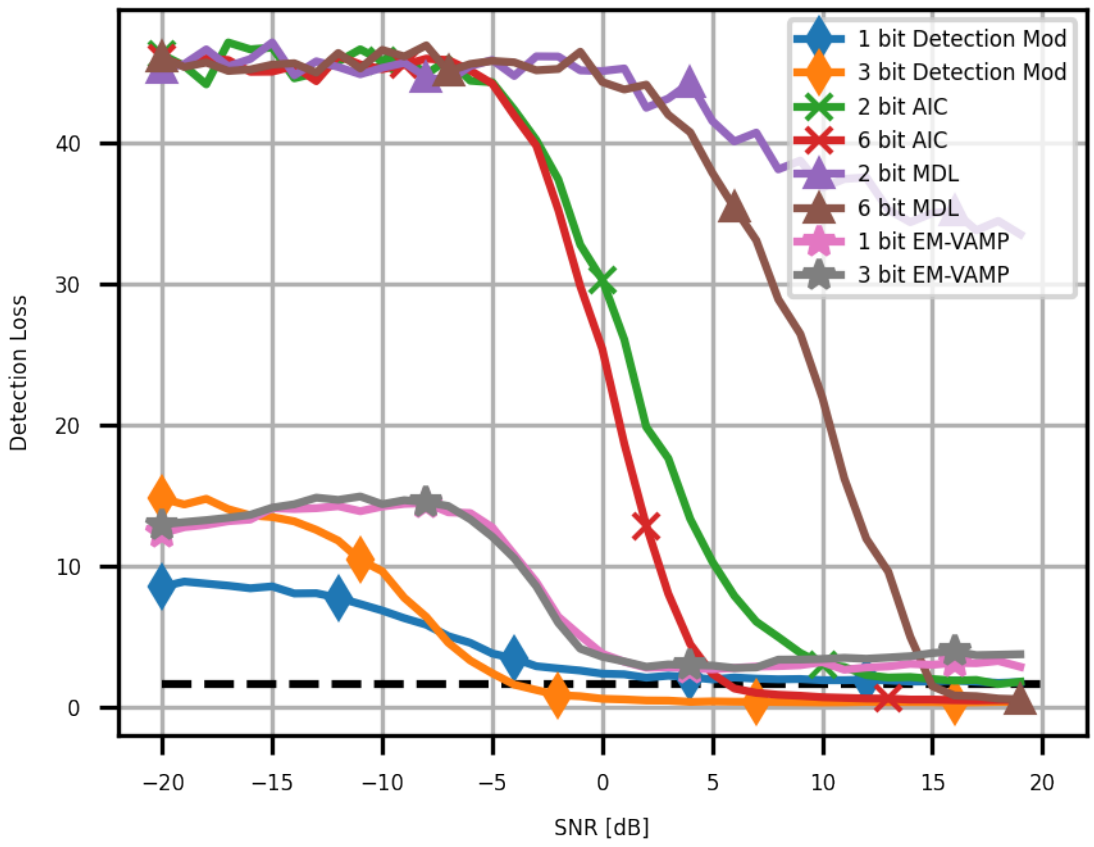


Figure 8.1: Our detection module compared to EM-VAMP, MDL and AIC with the proposed dtection loss metric. We double the data resolution for MDL and AIC so that the results are more comparable. It can be seen that the one-bit detection module is unable to surpass the learning threshold, thus never learning meaningful results from the input. The threshold is also far below either MDL or AIC until 7dB, showing the inherent advantage data driven techniques have, even for uninformative output distributions like a discrete uniform.

8.2 Frequency estimation

Next, we evaluate the frequency estimation performance of the estimation module. The distribution of the output vector $\mathbf{f}$ comes from both the initial w_0 and the offsets, w_i. While the

compared algorithms must search for each point with no prior, the actual frequency distributions have structure that can be exploited for better estimation at low SNR. This is one reason why the Periodogram estimator performs significantly worse than the neural architecture in Figure 8.2, and does not reach the threshold for SNR < -5dB. Following Algorithm 1, the learning threshold is calculated for each number of sinusoidal components shown by assuming m is known and computing the expected mean squared error. The estimator, $g_\theta(\mathbf{X})$ is simply the mean estimator because the loss function is the mean squared error, exactly as shown in Chapter 3. Note that the calculations are carried out in linear scale, although the plotting is done in dB scale.

First, we note the first and second moments of the folded normal distribution, used for the frequency components beyond the initial one, and approximated for our simulation settings

$$\mathbb{E}[w_i] = \sqrt{\frac{5}{N\pi}} \approx 0.1577 \quad i \in \{1, ... \ m\} \tag{8.13}$$

$$\sigma^2_{w_i} = \frac{2.5}{N} - E[w_i]^2 \ = \ \frac{5}{2N}(1 - \frac{2}{\pi}) \approx 0.0142. \tag{8.14}$$

Based on these statistics and similar measures associated with uniform variables, we can get the threshold and constant vector estimator for the frequency estimation results

$$g_\theta(\mathbf{X}) = \left[\mathbb{E}[w_0], ..., \ \mathbb{E}[w_0] + \frac{m-1}{N} + \mathbb{E}[w_m] \right] \tag{8.15}$$

$$= \left[0.125, \ ..., \ 0.125 + \frac{(m-1)}{N} + \sqrt{\frac{5}{N\pi}} \right]. \tag{8.16}$$

The independence of w_i is used in (8.15) to separate the expectations. Because $g_\theta(\mathbf{X})$ is an unbiased estimator, the threshold only depends the variance of the random vector f_i. We use $\sigma^2()$ to

refer to the variance of a random variable, so the threshold is

$$\mathrm{L}_{\text{threshold},\mathbf{f}}(m) = \frac{1}{m}\sum_{i=1}^{m}\sigma^2(f_i) \tag{8.17}$$

$$= \frac{1}{m}\sum_{i=1}^{m}\frac{1}{64} + \frac{1}{m}\sum_{i=2}^{m}\frac{5}{2N}\left(1-\frac{2}{\pi}\right) \tag{8.18}$$

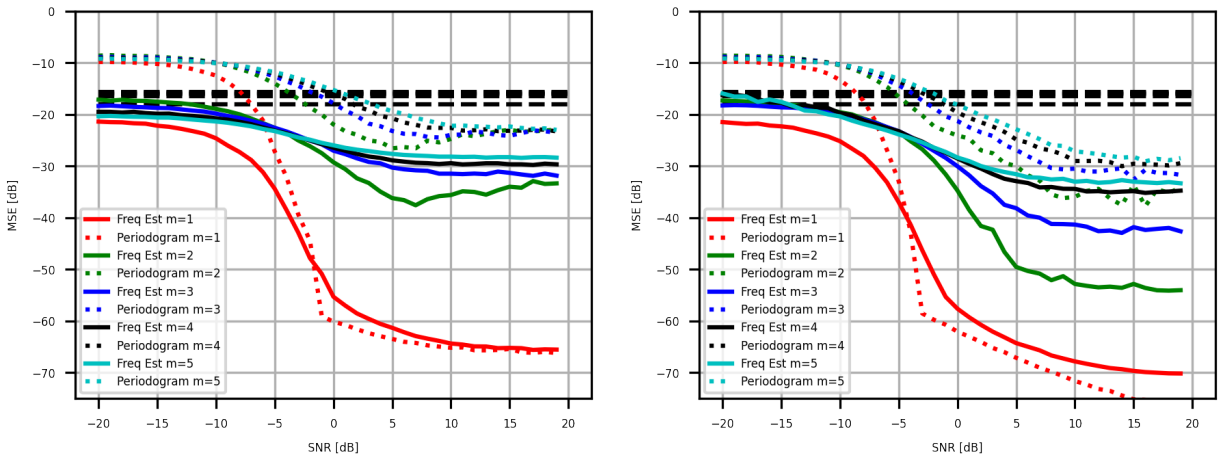
$$= \frac{1}{64} + \left(1-\frac{1}{m}\right)\frac{5}{2N}\left(1-\frac{2}{\pi}\right). \tag{8.19}$$

Here we average over all samples and outputs for multiple output mean squared error to produce a scalar loss value. This results in $\mathrm{L}_{\text{threshold},\mathbf{f}} \approx [-18\text{dB}, -16.4\text{dB}, -16\text{dB}, -15.8\text{dB}, -15.7\text{dB}]$ for each value of $1 \leq m \leq M$. While there is potential for the max-cutoff at $f = 0.5$ to cause the distributions to be heavy-tailed (see e.g. Fig. 5.1-5.2), this does not occur in expectation, so we do not include it in the simplified learning threshold. It can be seen, however, that it does have an effect on the distributions for $m = \{4, 5\}$ in Figure 5.1.

Our results show that, especially for $m \in \{2, 3\}$, our algorithm consistently outperforms the Periodogram by $3 - 8$dB, and is only surpassed for the $m = 1$ case with SNR > -4dB. Given more data, it is likely that our algorithm would reach or surpass the Periodogram consistently, however we only train on a small dataset to focus on the insights and learning of our architecture. Our results are especially interesting, because the potential gain of using our algorithm for two- or three-sinusoid signals is an order of magnitude better performance.

8.3 Amplitude estimation

Next, we consider the amplitude estimates of our algorithm. While the frequency estimates are resolvable for any bit resolution with sufficiently many samples N and reasonable spacing, the amplitude estimates can be particularly challenging near critical frequencies [28]. For example,



(a) one-bit

(b) three-bit

Figure 8.2: Two plots of the results of our sinusoidal estimation module's frequency outputs (Freq Mod –) for a given m over SNR and for one-and three-bit data. Our estimator outperforms the threshold universally, and the periodogram (...) except in the $m = 1$ case for SNR ≥ -4dB. For $m > 1$, our estimator is consistently better than the periodogram. In (a) both our networks and the Periodogram experience worse performance for high SNR, which we also observed in our past work with one-bit quantization [2]. Our results only show the performance loss for $m = 2$, while the Periodogram converges to $L = -23$dB for all $m > 1$. The results in (b) no longer show regressing performance, and the increased data resolution especially improves the $m = 2, 3$ cases, where our estimator improves upon the Periodogram by 11-15dB.

consider the following: a single sinusoid with normalized frequency of $f = 0.25 \pm \epsilon$, where $\epsilon < 1/N$, $\phi = 0.1$, and one-bit quantization. In this setting, every subset of $1/f = 4$ points are identical and only contain $\{1+j, -1+j, -1-j, 1-j\}$. Thus the received sequence, without noise, will be $N/4$ repetitions of that sequence, which makes amplitude recovery extremely difficult. While this is true for frequency estimation as well, the repetition improves the frequency estimate, and as $N \to \infty$ the frequency estimate will converge to f. There is no similar guarantee for the amplitude estimates, because regardless of the amplitude, the observation sequence is still the

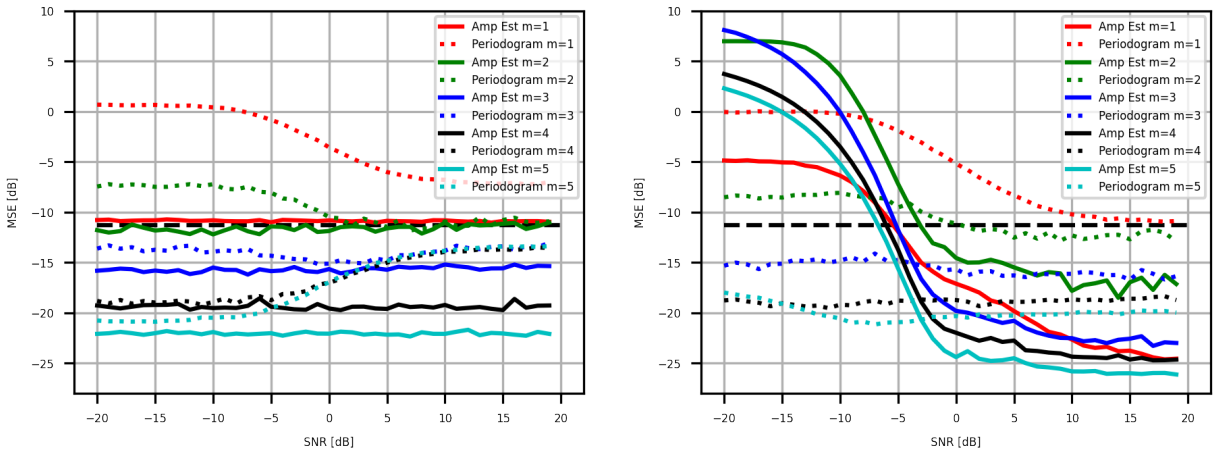
same. Even with infinite SNR very little amplitude information is recoverable from such limited data. Nevertheless, the learning threshold does not depend on the input data, so we can still analyze the expected worst case non-adversarial results. Because a_i are independent, the threshold is simply

$$\begin{aligned}\mathbb{E}[a_i] &= 0.55 \\ L_{\text{threshold,a}}(m) &= \frac{1}{m}\sum_{i=1}^{m}\sigma^2(a_i) \\ &= \sigma^2(a_1) = 0.0675.\end{aligned}$$

While the amplitude learning threshold is on a similar scale to the frequency learning threshold, actually achieving this value is difficult for low resolution due to quantization effects directly impacting the amplitude of the signal. We will again rely on the Periodogram as a benchmark for amplitude estimation, although it is not known whether other algorithms have been shown to be more effective for quantized sinusoids.

It can be seen in Figure 8.3a that our algorithm is not able to successfully surpass the threshold with one-bit data for $m < 3$. Compared with the frequency results from Figure 8.2a, the one-bit amplitude estimates are not a significant improvement over the threshold or benchmark, and the improvements are entirely coming from memorizing the distribution, rather than learning useful features within the data. This is visible by the lack of improvement with increasing SNR. Increasing the bit resolution to three-bits provides a significant improvement for SNR > -5dB, suggesting that meaningful information is being used to recover the amplitudes and surpass the Periogram in every case.

From Figure 8.3, we obtain insight into what is happening between the one- and three-bit versions. In 8.3a most of the estimators trend are not learning any meaningful features from the input and are simply minimizing the loss over the distribution. In 8.3b, the estimators perform



(a) (b)

Figure 8.3: A comparison of our sinusoid estimator's amplitude results (Amp Est –). Unintuitively, our estimator appears to perform better for one-bit data than three-bit for low SNR. Identifying the learning threshold, however, shows the Periodogram and our $m = \{1, 2\}$ estimators converging to a similar level, thus learning nothing of importance from $\mathbf{X}$ in (a). In contrast, all of our estimators surpass the threshold for three-bit data in (b).

worse initially because of the quantization and noise effects, but outperform the threshold for SNR > -3dB. Thus our three-bit results are much more useful and generalizable than the one-bit case, even if the performance is initially worse for low SNR. From these plots in particular, we can see the value of defining the learning threshold when evaluating deep learning algorithms. Interestingly, increasing the number of sinusoids to $m = \{3, 4, 5\}$ leads to better results, even though the true amplitudes are independent. We believe this is because as m increases, the average error tends toward the arithmetic mean of each the amplitude estimates, but in the case of $m = \{2, 3\}$, if the minimum separation is not sufficient, the estimation error is dominated by the overlapping components.

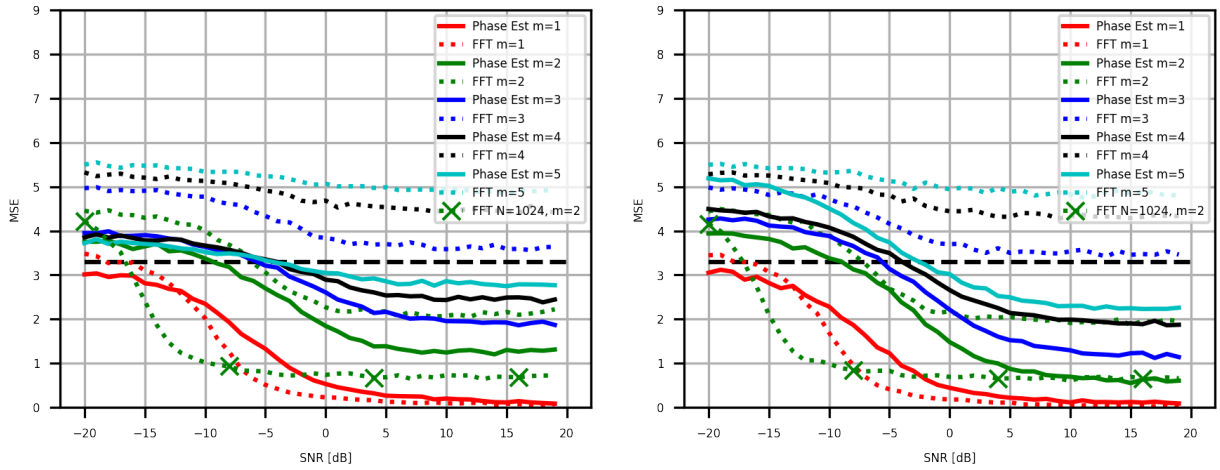
8.4 Phase estimation

In the final evaluation of the sinusoid estimator, we look at the phase estimates. We begin in a similar fashion to the amplitude and phase results by first solving for the learning threshold. Similar to the amplitude distribution, the phases are uniformly distributed, $\phi_i \in [0, 2\pi]$, so the constant estimator and learning threshold are again simple to compute,

$$\begin{aligned}\mathbb{E}[\phi_i] =& \pi \\ \mathrm{L}_{\text{threshold},\boldsymbol{\phi}}(m) =& \frac{1}{m}\sum_{i=1}^{m}\sigma^2(\phi_i) \\ =& \sigma^2(\phi_1) = \frac{\pi^2}{3}.\end{aligned}$$

From the learning thresholds, we expect the phase error to be significantly higher than the other estimator losses, which is why we scale the sum training loss according to the learning thresholds in Chapter 5. This should help ensure none of the losses affect the model significantly more than the others, based on the simulation setup.

As mentioned in Chapter 6, the Periodogram is replaced with the FFT as a benchmark for estimating the phase of a signal. We show the results of this algorithm with ours in Figure 8.4, this time without dB scaling. We can see that the one-bit results in Fig. 8.4a are better than the amplitude estimates, in the sense that every estimator is able to learn and improve with increasing SNR. Similar to the results from Figure 8.2b, only the $m = 1$ case shows the benchmark (FFT) performing better than our estimator. In general, the FFT does not perform well because of the short sample lengths $N = 64$, rather than the quantization. In the three-bit results, Fig. 8.4b, we see that the FFT results are largely unchanged from the one-bit version, but our estimator now shows a noticeable improvement with increasing SNR. We also show the case where we increase



(a) (b)

Figure 8.4: A comparison of the estimation performance of our estimator and the FFT for (a) one-bit data and (b) three bit data. The chart on the left shows all of the estimators achieving and passing the threshold beyond SNR $= -5$dB, however most of the estimators have only a slight improvement over the threshold. In contrast, our estimator in (b) is able to successfully surpass the threshold for SNR > -3dB for every value of m. Interstingly, the FFT does not noticeably improve with the data resolution, however we found that this was due to the length $N = 64$ causing the phase estimates to be too coarse for the input data. To validate our suspicion, we show the $m = 2$ case with $N = 1024$ for the FFT as well. We can see that in (b) this phase estimate is *slightly* improved from (a), and significantly improved from the $N = 64$ setting.

$N \to 1024$ for the two-sinusoid FFT to show that quantization is not the limiting factor in the performance.

8.5 SignalNet results

In the final results, we join the two modules, and consider pairings with different traditional estimators. We no longer show the learning threshold, instead comparing across different combi-

nations of algorithms to see how effective our algorithm is in the overall performance. We start by demonstrating the effects of normalization on SignalNet. As we previously found, the phase estimates tend to have much higher error, simply because the phase values are drawn from a distribution with a higher variance. Without normalization, the loss comparison, in dB, would look like the dotted results (. . .) in Figure 8.5. We can conclude that the loss landscape without scaling could result in neural networks that are almost entirely optimized by phase estimates, even though the loss–relative to the output randomness and effect on the input–is not significant. Because the Chamfer loss, assuming the correct number of sinusoids ($m = \hat{m}$), is just two times the mean absolute error, we can approximate the thresholds by the root mean squared error, which is an upper bound on the mean absolute error. Thus, we normalize by the square root of the learning thresholds from Chapter 8.

After normalizing, we evaluate benchmarks using AIC with the Periodogram/FFT against SignalNet, because AIC had much more success in quantized detection in Figure 8.1. We interchange AIC with our detection module as well and interchange the Periodogram/FFT with our sinusoidal estimator to identify which components provide the most gain in Figure 8.6. We color in the two regions corresponding to the gain from using our sinusoidal estimator (purple shading) and our detection module (red shading).

From Figure 8.5 we can see that normalizing by the learning threshold also provides a useful metric for identifying the learning occurring from the input data. By normalizing out the learning threshold, we can see what information can be gained from the input data specifically. The phase estimates in particular, are able to benefit from the input data, while the amplitude estimates had the least improvement beyond the threshold. While this can be expected based on intuition regarding amplitude quantization and the results from Figures 8.3a-8.3b, it further validates the

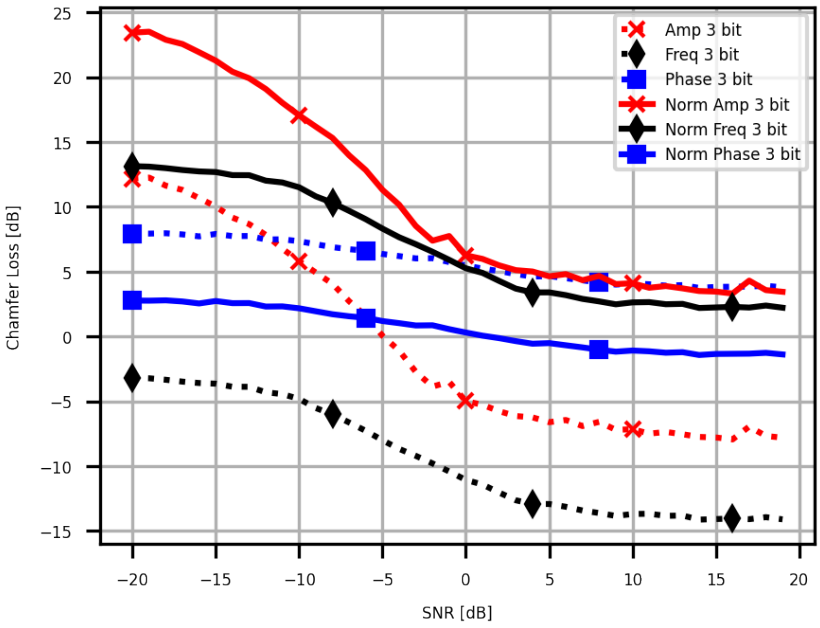
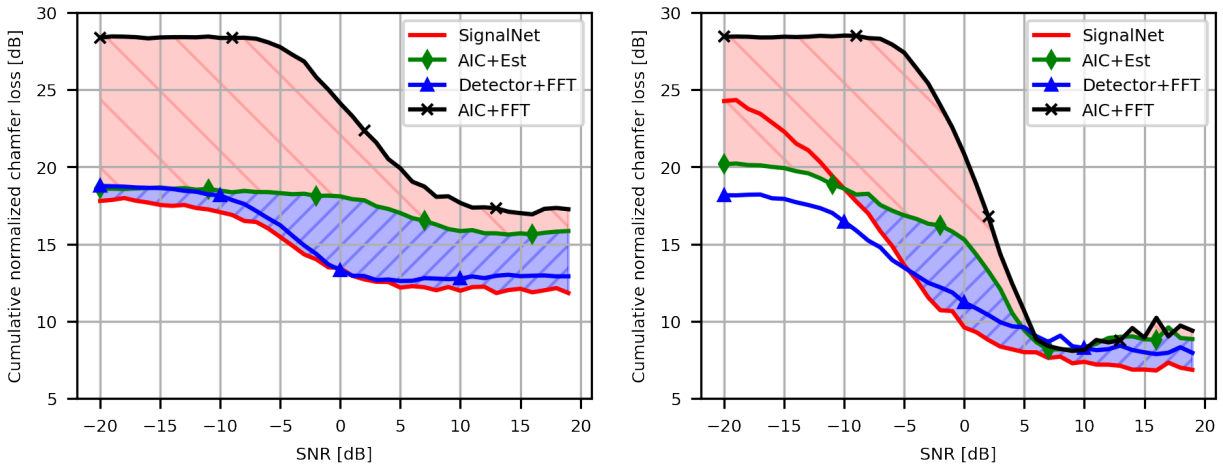


Figure 8.5: The comparison for three-bit estimators with (. . .) and without (–) normalization. We can see the performance order switches from (Freq < Amp < Phase) to (Phase < Freq < Amp).

use of the learning threshold for normalization.

The final results in Fig. 8.6, show the sinusoid estimation module providing the most noticeable gain. We can also see that the combination is not uniform: switching to our detection module from AIC provides significant gain, but so does switching to our sinusoid estimator from the Periodogram/FFT. This is because our sinusoid estimator has learned to fit the best estimates for a given number of sinusoids, so even with poor detection results, it still has reasonable Chamfer loss. This explains the asymmetric gain from adding or subtracting one of our modules between the AIC+FFT curve and our SignalNet curve. Thus, we can see that applying machine learning, either at the detector or the estimator, provides significant gain. Recall however, that the learning



(a) (b)

Figure 8.6: The resulting chamfer loss for each of the algorithms with one-bit data (a) and three-bit data (b). We color two regions: one region to show the gain from our detector (blue //) and the other to show the gain from our sinusoid estimator (red \\). The gain is primarily due to our sinusoid estimator because it is trained to perform the best estimates for a specific value of m, so even if the AIC detection is incorrect, the estimates will all be reasonably close to the true values.

threshold normalization has been applied to the estimation module, but not to the detection module, as there is no clear way to apply that with the Chamfer loss definition. Based on the results from the one-bit detection module in Fig. 2.1, a significant portion of the gain being seen by the detection module can be attributed to learning the distribution, rather than learning that generalizes beyond our simulation. Ultimately, we can conclude that applying our SignalNet architecture to the detection and estimation task is a valuable improvement over other techniques.

Chapter 9

Conclusion

In this paper, we described and evaluated SignalNet, a novel deep neural network for multi-sinusoidal decomposition from low resolution sampling. While no other work has considered quantized, multi-sinusoidal decomposition, our network follows traditional, unquantized work by dividing the problem into the subtasks of detection and estimation. We describe a distinct architecture for each subtask, and specifically focus on developing a novel estimation network. Our estimation network uses internal reconstruction to explicitly learn the input-output relationship, leading to generally successful sinusoidal estimates for low resolution data. Our algorithm struggled to learn from one-bit data in both detection and amplitude estimation though, similar to the baseline algorithms. We observe that recovering amplitude information and detecting the number of sinusoids in a signal is extremely difficult for short sequences of one bit data. Yet, we also saw the benefit that instilling domain knowledge, reconstructive knowledge and phase-frequency relationships, provided to our estimator, allowing it to efficiently learn important features from the data.

We also proposed a theoretical tool for comparing our networks and normalizing across distributions by defining a learning threshold. The threshold is used to express the randomness in the output variable for a specific loss function. We analyzed our simulation settings and found the learning thresholds for each subtask within our network based on statistical estimation theory. We used the insight from the learning threshold to understand why the performance of our model

did not universally increase going from the one-bit to the three-bit versions, and often had better low SNR performance with one-bit data–by not learning meaningful features. This prevented the estimator from performing well, but also made it robust to noise. We found that our learning threshold is a useful metric to consider when comparing neural networks with broader algorithms and determining whether learning has been successful.

Finally, we combined our networks together to complete the SignalNet architecture. Our unified network is able to surpass the benchmark algorithms universally. In comparing the Chamfer loss, we found that our detection module provides a significant improvement over AIC, but without normalization it was unobvious how much more effective it is in a more general setting. We were able to directly quantify the improvement from our sinusoid estimator though, which provided between 2-10dB improvement. We conclude that a domain-aware detection module could improve the results further, as well as additional investigation into loss functions like Chamfer loss, that can be weighted to encourage over estimating, like our detection module was trained on. Our results suggest that multi-sinusoid decomposition can be performed even with extremely low resolution quantization using our SignalNet architecture.

Bibliography

[1] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *NeurIPS Deep Learning and Representation Learning Workshop*, 2015. [Online]. Available: http://arxiv.org/abs/1503.02531

[2] R. M. Dreifuerst, R. W. Heath, M. N. Kulkarni, and J. Zhang, "Deep learning-based carrier frequency offset estimation with one-bit ADCs," in *Signal Processing Advances in Wireless Communications (SPAWC)*, 2020, pp. 1–5.

[3] L. G. Beatty, J. D. George, and A. Z. Robinson, "Use of the complex exponential expansion as a signal representation for underwater acoustic calibration," *The Journal of the Acoustical Society of America*, vol. 63, no. 6, pp. 1782–1794, 1978.

[4] A. Ghasemi and E. S. Sousa, "Spectrum sensing in cognitive radio networks: requirements, challenges and design trade-offs," *IEEE Communications Magazine*, vol. 46, no. 4, pp. 32–39, 2008.

[5] J. Capon, "High-resolution frequency-wavenumber spectrum analysis," *Proceedings of the IEEE*, vol. 57, no. 8, pp. 1408–1418, 1969.

[6] D. Cabric, S. M. Mishra, and R. W. Brodersen, "Implementation issues in spectrum sensing for cognitive radios," in *Asilomar Conference on Signals, Systems and Computers*, vol. 1, 2004, pp. 772–776 Vol.1.

[7] R. W. Heath, N. Gonzalez-Prelcic, S. Rangan, W. Roh, and A. M. Sayeed, "An overview of signal processing techniques for millimeter wave MIMO systems," *IEEE J. Select. Areas Commun.*, vol. 10, no. 3, pp. 436–453, April 2016.

[8] Y. Zhou and W. Yu, "Optimized backhaul compression for uplink cloud radio access network," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1295–1307, 2014.

[9] J. Liu, Z. Luo, and X. Xiong, "Low-resolution ADCs for wireless communication: A comprehensive survey," *IEEE Access*, vol. 7, pp. 91 291–91 324, 2019.

[10] J. J. Bussgang, "Crosscorrelation functions of amplitude-distorted Gaussian signals," 1952. [Online]. Available: {http://hdl.handle.net/1721.1/4847}

[11] O. T. Demir and E. Bjornson, "The Bussgang decomposition of nonlinear systems: Basic theory and MIMO extensions," *IEEE Signal Processing Magazine*, vol. 38, no. 1, pp. 131–136, 2021.

[12] A. Fredriksen, D. Middleton, and V. VandeLinde, "Simultaneous signal detection and estimation under multiple hypotheses," *IEEE Transactions on Information Theory*, vol. 18, no. 5, pp. 607–614, 1972.

[13] M. Wax and T. Kailath, "Detection of signals by information theoretic criteria," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 33, no. 2, pp. 387–392, 1985.

[14] M. Wax and I. Ziskind, "Detection of the number of coherent signals by the MDL principle," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, no. 8, pp. 1190–1196, 1989.

[15] P. Stoica, R. L. Moses, B. Friedlander, and T. Soderstrom, "Maximum likelihood estimation of the parameters of multiple sinusoids from noisy measurements," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, no. 3, pp. 378–392, 1989.

[16] S. C. A. Thomopoulos and T. W. Hilands, "Joint detection/estimation for modal order selection in Gaussian and nonGaussian noise," in *MILCOM 92 Conference Record*, 1992, pp. 613–617 vol.2.

[17] P. M. Djuric, "Simultaneous detection and frequency estimation of sinusoidal signals," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. 4, 1993, pp. 53–56.

[18] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, pp. 465–471, 1978.

[19] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Proc of the 2nd International Symposium on Information Theory*, 1973, pp. 267–281.

[20] R. O. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. Antennas Propagat.*, vol. 34, no. 3, pp. 276–280, March 1986.

[21] M. H. Hayes, *Statistical Digital Signal Processing and Modeling*, 1st ed. John Wiley and Sons, Inc., 1996.

[22] D. L. Donoho, A. Maleki, and A. Montanari, "Message-passing algorithms for compressed sensing," *Proc. of the National Academy of Sciences*, vol. 106, no. 45, pp. 18 914–18 919, 2009.

[23] S. Rangan, "Generalized approximate message passing for estimation with random linear mixing," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, 2011.

[24] A. K. Fletcher and P. Schniter, "Learning and free energies for vector approximate message passing," in *Proc. IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[25] G. Izacard, S. Mohan, and C. Fernandez-Granda, "Data-driven estimation of sinusoid frequencies," in *Adv. Neural Inf. Process. Syst.*, Jun. 2019, pp. 5127–5137.

[26] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[27] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," 2017.

[28] A. Høst-Madsen and P. Händel, "Effects of sampling and quantization on single-tone frequency estimation," *IEEE Trans. Signal Processing*, vol. 48, no. 3, pp. 650–662, 2000.

[29] O. Dabeer and A. Karnik, "Signal parameter estimation using 1-bit dithered quantization," *IEEE Trans. Inform. Theory*, vol. 52, no. 12, pp. 5389–5405, Dec. 2006.

[30] C. Gianelli, L. Xu, J. Li, and P. Stoica, "One-bit compressive sampling with time-varying thresholds: Maximum likelihood and the Cramér-Rao bound," in *Proc. of Asilomar Conf. on Signals, Systems and Computers*, 2017, pp. 399–403.

[31] H. Nyquist, "Certain topics in telegraph transmission theory," *Transactions of the American Institute of Electrical Engineers*, vol. 47, no. 2, pp. 617–644, 1928.

[32] A. Wadhwa and U. Madhow, "Blind phase/frequency synchronization with low-precision adc: A bayesian approach," in *Allerton Conference on Communication, Control, and Computing (Allerton)*, 2013, pp. 181–188.

[33] M. Abdizadeh, J. T. Curran, and G. Lachapelle, "New decision variables for gnss acquisition in the presence of cw interference," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 50, no. 4, pp. 2794–2806, 2014.

[34] G. Zeitler, G. Kramer, and A. C. Singer, "Bayesian parameter estimation using single-bit dithered quantization," *IEEE Trans. Signal Process.*, vol. 60, no. 6, pp. 2713–2726, jun 2012.

[35] J. Mo, P. Schniter, and R. W. Heath, "Channel estimation in broadband millimeter wave MIMO systems with few-bit ADCs," in *IEEE Trans. Signal Process.*, vol. 66, no. 5, mar 2018, pp. 1141–1154.

[36] S. Jacobsson, G. Durisi, M. Coldrey, U. Gustavsson, and C. Studer, "Throughput analysis of massive mimo uplink with low-resolution adcs," *IEEE Transactions on Wireless Communications*, vol. 16, no. 6, pp. 4038–4051, 2017.

[37] B. G, "Distance transformations in arbitrary dimensions," *Computer Vision, Graphics, and Image Processing*, vol. 27, pp. 321–345, 1984.

[38] T. Cover, "Broadcast channels," *IEEE Transactions on Information Theory*, vol. 18, no. 1, pp. 2–14, 1972.

[39] R. M. Dreifuerst. [Online]. Available: https://github.com/Ryandry1st/SignalNet

[40] E. J. Candès and C. Fernandez-Granda, "Towards a Mathematical Theory of Super-resolution," *Communications on Pure and Applied Mathematics*, vol. 67, no. 6, pp. 906–956, 2014.

[41] A. Moitra, "Super-Resolution, Extremal Functions and the Condition Number of Vandermonde Matrices," in *Proc. of ACM Symposium on Theory of Computing*, 2015, p. 821–830.

[42] S. Jacobsson, C. Lindquist, G. Durisi, T. Eriksson, and C. Studer, "Timing and frequency synchronization for 1-bit massive MU-MIMO-OFDM downlink," in *Signal Processing Advances in Wireless Communications (SPAWC)*, 2019, pp. 1–5.

Vita

Ryan obtained his B.S. in Electrical Engineering from Milwaukee School of Engineering with minors in Mathematics and Physics. He also received his B.S. in Electrical and Communications Engineering from Technische Hochschule Lübeck. He is pursuing his M.S. and Ph.D. in electrical engineering at The University of Texas at Austin (UT Austin). His research lies at the interchapter of machine learning and signal processing. Specifically, he has focused on augmenting machine learning with domain knowledge for physical layer processing in wireless communications.

Permanent address: 710 Franklin Blvd
Austin, Texas 78751